09/830588

PCT/JP00/05007

01.08.00

日 本 国 特 許 庁
PATENT OFFICE
JAPANESE GOVERNMENT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日
Date of Application: １９９９年 ９月 ９日

出 願 番 号
Application Number: 平成１１年特許願第２５５２７２号

出 願 人
Applicant (s): 科学技術振興事業団

２０００年 ９月 １日

特 許 庁 長 官
Commissioner,
Patent Office 及川耕造

出証番号 出証特２０００－３０６９１２３

【書類名】　特許願

【整理番号】　P0159JP

【提出日】　平成11年 9月 9日

【あて先】　特許庁長官殿

【国際特許分類】　G06F　9/00

【発明者】

【住所又は居所】　千葉県千葉市稲毛区弥生町１－１７０－２－２０３

【氏名】　松本　尚

【特許出願人】

【識別番号】　396020800

【氏名又は名称】　科学技術振興事業団

【代理人】

【識別番号】　100107010

【弁理士】

【氏名又は名称】　橋爪　健

【手数料の表示】

【予納台帳番号】　054885

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【プルーフの要否】　要

【書類名】　　明細書

【発明の名称】　アクセス方法及びアクセス処理プログラムを記録した記録媒体

【特許請求の範囲】

【請求項１】

第１のプロセスが、入出力装置又はインタフェースのオープン処理をオペレーティングシステムに要求するステップと、

オペレーティングシステムが、第１のプロセスの要求格納領域を示すためのコンテクスト識別子を割り当て、該コンテクスト識別子に該当するメモリページを未処理の要求があることを示すペンディングレジスタのアクセス用アドレスとしてマップするステップと、

第１のプロセスが、要求格納領域へ入出力装置又はインタフェースへの要求内容を記述するステップと、

オペレーティングシステムが、ペンディングレジスタのアクセス用アドレスを使って、未処理の要求があることを入出力装置又はインタフェースに伝えるステップと、

入出力装置又はインタフェースが、ペンディングレジスタに記憶されたコンテクスト識別子に基づき、第１のプロセスの要求を読み出すステップ

を含むアクセス方法。

【請求項２】

オペレーティングシステムは、入出力装置又はインタフェース内の内蔵メモリに、各プロセスに対応する要求格納領域の物理アドレスを記憶するようにしたことを特徴とする請求項１に記載のアクセス方法。

【請求項３】

入出力装置又はインタフェースは、

ペンディングレジスタにアクセスがあったことを識別するステップと、

コンテクスト識別子に基づき、各プロセスの要求格納領域の物理アドレスを記憶した内蔵メモリを参照することにより、ペンディングレジスタをアクセスしたプロセスの要求格納領域の物理アドレスを得るステップと、

要求格納領域の内容を読み出して、要求内容を実現するステップと

を備えたことを特徴とする請求項１又は２に記載のアクセス方法。

【請求項４】

入出力装置又はインタフェースを指定する物理アドレスは、ペンディングレジスタの位置を示すファンクションセレクト領域と、プロセスを示すコンテクスト識別子領域とを含み、

アドレスデコーダは、前記物理アドレスのファンクションセレクト領域にペンディングレジスタの場所を表す固有アドレスが記憶されている場合、前記物理アドレスのコンテクスト識別子をペンディングレジスタに記憶することを特徴とする請求項１乃至３のいずれかに記載のアクセス方法。

【請求項５】

前記ペンディングレジスタに、コンテクスト識別子に対応する他のデータを必要に応じて記憶することを特徴とする請求項１乃至４のいずれかに記載のアクセス方法。

【請求項６】

第１のプロセスが入出力装置又はインタフェースのクローズ処理をオペレーティングシステムに要求した場合又は第１のプロセスが終了した場合には、オペレーティングシステムは第１のプロセスに割り当てたアドレスを回収すると共に、入出力装置又はインタフェース用のコンテクスト識別子を回収し、及び又は、内蔵メモリ上の第１のプロセスの要求領域への物理アドレスのエントリをクリアすることを特徴とする請求項１又は５のいずれかに記載のアクセス方法。

【請求項７】

第１のプロセスが、入出力装置又はインタフェースのオープン処理をオペレーティングシステムに要求するステップと、

オペレーティングシステムが、第１のプロセスの要求格納領域を示すためのコンテクスト識別子を割り当て、該コンテクスト識別子に該当するメモリページを未処理の要求があることを示すペンディングレジスタのアクセス用アドレスとしてマップするステップと、

第１のプロセスが、要求格納領域へ入出力装置又はインタフェースへの要求内容を記述するステップと、

オペレーティングシステムが、ペンディングレジスタのアクセス用アドレスを使って、未処理の要求があることを入出力装置又はインタフェースに伝えるステップと、

入出力装置又はインタフェースが、ペンディングレジスタに記憶されたコンテクスト識別子に基づき、第１のプロセスの要求を読み出すステップ

を含むアクセス処理プログラムを記録した記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、アクセス方法及びアクセス処理プログラムを記録した記録媒体に係り、特に、仮想化されたマルチコンテクスト環境におけるユーザレベル入出力アクセス方法及びアクセス処理プログラムを記録した記録媒体に関する。

【０００２】

【従来の技術】

図６に、従来方式１のシステムの説明図を示す。この図では、従来のカーネルアドレス空間にマップされたシステムの説明図を示す。

【０００３】

従来方式１では、図示のように、入出力装置（Ｉ／Ｏ装置）(I/O device)の制御用レジスタは、汎用オペレーティングシステム（ＯＳ）の下では、カーネル（kernel）モードでしかアクセスできないようにすることにより、ユーザアプリケーションによる不正な操作を防止している。物理アドレス空間（physical address space）は、カーネルアドレス空間(kernel address space)とユーザアドレス空間(user address space)とに分かれる。Ｉ／Ｏ装置は、カーネルアドレス空間のＩ／Ｏ装置に対応するエントリがアクセスされた場合のみ、Ｉ／Ｏ装置を使用することができる。そして、ユーザアプリケーションとしてのプロセスＡが、Ｉ／Ｏ装置を使用したい場合には、ＯＳのシステムコール(system call)を通じて操作要求を出す。つぎに、ＯＳが、カーネルモードにおいて操作要求の正当性をチェックした後、具体的な操作をＩ／Ｏ装置内の制御レジスタに施す。

【０００４】

また、最近では、ユーザアプリケーションのアドレス空間にＩ／Ｏ装置の制御レジスタをマップして、ユーザアプリケーション（ユーザプロセス）が直接制御レジスタをアクセスし、入出力操作のオーバヘッドを減らす方式も使われるようになってきている。図７に、従来方式２のシステムの説明図を示す。この図では、ユーザアドレス空間にマップされたシステムの構成図の一例を示す。この従来方式２では、プロセスＡが、物理アドレス空間にあるＩ／Ｏ装置用ページ(page for I/0)を、ユーザアドレス空間にマッピングしている。Ｉ／Ｏ装置用ページにより、Ｉ／Ｏ装置の制御レジスタを使用することができる。

【０００５】

つぎに、図８に、従来方式３のシステムの説明図を示す。この図は、ユーザアドレス空間から操作要求をダイレクトメモリアクセス（ＤＭＡ）するシステムの一例を示す。従来方式１におけるシステムコールのオーバヘッドを排除又は小さくするためには、ユーザアドレス空間にＩ／Ｏ装置に関する制御情報をマップする必要がある。従来方式３においても、この点では従来方式２と同じである。ただし、単にＩ／Ｏ装置の制御レジスタをユーザアドレス空間にマップしたのでは、複数のユーザアプリケーションからアクセスされた場合に、混乱が生じてしまう。これを解決するためには、Ｉ／Ｏ装置に対する要求内容をメモリ上にまとめておいて、Ｉ／Ｏ装置側に要求内容を読み出して制御レジスタにセットしてもらうようにすればよい。つまり、Ｉ／Ｏ装置側がＤＭＡによって制御レジスタの内容を更新する。そして、一つの要求内容が終る（区切りがつく）までは、次の要求内容の処理を開始しないようにＩ／Ｏ装置が動作すれば動作の混乱も起こらない。また、汎用ＯＳの下でメモリ資源は保護されているので、アプリケーション間で要求内容を書き潰したりする心配はない。

【０００６】

ここで重要になるのは、どうやってＩ／Ｏ装置に要求内容が書かれているメモリの場所を指示するかである。そこで、従来方式３では、要求内容を示すメモリの場所を示すポインタを記憶したコンテクストレジスタ(context register)をＩ／Ｏ装置内に持たせる。コンテクストレジスタは、例えば、どのプロセス・タスクがホストプロセッサで実行されているかを示すレジスタである。ＯＳは、どの

ユーザのタスク（プロセス）が実行しているかを把握している。そして、ＯＳは、実行されているプロセスに応じて、コンテクストレジスタを書き換えることができる。プロセッサにあるアプリケーション（例えば、プロセスＡ）が割り当てられる場合には、該当アプリケーションの要求内容の格納場所をコンテクストレジスタにセットする。

【０００７】

また、各アプリケーションによるユーザアドレス空間には、要求内容を登録したことを知らせるペンディングレジスタ(ペンディングレジスタ)をマップする。ペンディングレジスタは、例えば、いずれかのプロセスがＩ／Ｏ装置をアクセスしたことを示すフラグである。プロセスＡ用ページ(page for A)には、要求（requests）が記憶される。要求の内容は、例えば、Ｉ／Ｏ指令であり、Ｉ／Ｏ装置がプリンタであれば、印刷指令、改ページ指令等であり、入出力装置であれば、どの装置に出力又は入力するかの指令等である。また、従来方式３では、Ｉ／Ｏ装置はペンディングレジスタがアクセスされると、コンテクストレジスタの（ポインタの）示すメモリ位置から該当する要求内容を取り出して（ＤＭＡして）、実行する。

【０００８】

【発明が解決しようとする課題】

しかしながら、従来方式１は、ユーザアプリケーションが入出力操作をする必要がある毎に、システムコールを含む一連のカーネルプログラムを呼び出す必要があり、オーバヘッドが大きくなるという課題がある。

【０００９】

このため、従来方式２が提案されたものの、この従来方式２では、同時に複数のアプリケーションが該当Ｉ／Ｏ装置を使うことができない。この例では、プロセスＡがＩ／Ｏ装置を使用している場合に、プロセスＢがＩ／Ｏ装置を使用することができない。また、一つのアプリケーションが自分の都合だけでＩ／Ｏ装置の制御レジスタの内容を変更してしまうので、複数のアプリケーションが時分割で制御レジスタを操作すると、制御レジスタの操作の一貫性が崩れてしまう。よって、この従来方式２ではＩ／Ｏ資源を排他的に一つのアプリケーションに割り

当てて使用する必要がある。

【００１０】

そこで、従来方式１のシステムコールのオーバヘッドの課題、及び、従来方式２の割り当ての課題を解決する方式として従来方式３が提案された。しかしながら、従来方式３において、Ｉ／Ｏ装置内のコンテクストレジスタは、プロセッサコンテクストの一部となり、ＯＳがコンテクスト切替を行う毎に、プロセッサに割り当てられたアプリケーションの要求内容格納場所に更新する必要がある。コンテクスト切替は、ＯＳの根幹に関わる動作の一つであるため、既存のＯＳに対してプロセッサコンテクストを拡張する改造を施すことは難しい。ましてや、デバイスドライバの追加等では、プロセッサコンテクストは拡張できない。また、Ｉ／Ｏ装置が接続される計算機がマルチプロセッサ構成である場合にはコンテクストレジスタはプロセッサ台数と同じだけ必要であり、ペンディングレジスタへのアクセスがどのプロセッサからのアクセスであるか識別可能である必要がある。Ｉ／Ｏ装置の制御回路がＬＳＩ化される場合には、マルチプロセッサ対応を考慮すると、多くのコンテクストレジスタを用意しておく必要があり、通常の単体プロセッサもしくは少数台の並列システムでは無駄なコストになってしまう。

【００１１】

このように、従来行われているユーザレベルによるＩ／Ｏ装置へのアクセス方式はそれぞれ課題を抱えている。また、例えば、上述のように、通信およびＩ／Ｏ装置へのユーザレベルのアクセスを高速化するためにユーザメモリ空間に通信や入出力操作のためのレジスタをマップする手法が行われている。しかし、従来の方式では同時に複数のアプリケーションが使用することができなかった。

【００１２】

本発明は、以上の点に鑑み、メモリ管理機構を流用することにより、複数のアプリケーションが同時に低コストで通信および入出力操作、インタフェース操作を可能とするアクセス方法及びアクセス処理プログラムを記録した記録媒体を提供する。

【００１３】

【課題を解決するための手段】

本発明の特徴のひとつは、ページ単位のメモリエイリアス（物理的に同一の対象に複数のアドレスが割り当てられた状態）を故意に起こして、どのエイリアスアドレスからアクセスされたか検知する機構によってアクセスしたアプリケーションを同定することにある。メモリはページ単位に管理されているので、アプリケーションごとに別のエイリアスを割り当てることによって、他のエイリアスアドレスをアクセスするような不正な行動は行えない。及びアクセス処理プログラムを記録した記録媒体を提供する。

【００１４】

本発明の解決手段によると、

第１のプロセスが、入出力装置又はインタフェースのオープン処理をオペレーティングシステムに要求するステップと、

オペレーティングシステムが、第１のプロセスの要求格納領域を示すためのコンテクスト識別子を割り当て、該コンテクスト識別子に該当するメモリページを未処理の要求があることを示すペンディングレジスタのアクセス用アドレスとしてマップするステップと、

第１のプロセスが、要求格納領域へ入出力装置又はインタフェースへの要求内容を記述するステップと、

オペレーティングシステムが、ペンディングレジスタのアクセス用アドレスを使って、未処理の要求があることを入出力装置又はインタフェースに伝えるステップと、

入出力装置又はインタフェースが、ペンディングレジスタに記憶されたコンテクスト識別子に基づき、第１のプロセスの要求を読み出すステップ

を含むアクセス方法及びアクセス処理プログラムを記録した記録媒体を提供する。

【００１５】

【発明の実施の形態】

本発明では、上述の課題を解決する新しいユーザレベルによるＩ／Ｏ装置へのアクセス方法を提供する。

【００１６】

従来方式３において、入出力処理されていない要求がメモリ上にあることを示すペンディングレジスタへのアクセスが、どのアプリケーションによってなされたかを、Ｉ／Ｏ装置側が識別できれば、要求内容が格納されたメモリアドレスを求める方法が構成できる。極論すれば、コンテクストレジスタとペンディングレジスタの対が十分な数だけあり、アプリケーション空間ごとに一つのペンディングレジスタがマップされていれば、アクセスされたペンディングレジスタに対応したコンテクストレジスタの内容を使って、要求を読み出すアドレスを決定できる。この場合においてコンテクストレジスタをレジスタとして実装する必要はなく、Ｉ／Ｏ装置内のメモリ上の領域等で十分代用できる。アクセスされたペンディングレジスタの位置からメモリ上のコンテクストレジスタの位置を計算すればよい。コンテクストレジスタがＩ／Ｏ装置内のメモリ上の領域として実装可能であるなら、さらに、ペンディングレジスタの数を一つないしは少数に抑えることが可能になると、非常に低コストかつオーバヘッドのないＩ／Ｏ装置へのユーザレベルアクセス方法となる。

【００１７】

本発明では、ペンディングレジスタの実体は一つだけ用意するが、そこへのアドレスのエイリアスを複数用意し、どのエイリアスからアクセスされたか判る機構を用意することにより、ペンディングレジスタにアクセスしたアプリケーション（プロセッサコンテクスト）を同定する。以下に、本発明の実施の形態について、具体例を示して説明する。

【００１８】

図１に、本発明に関連するシステム構成図の一例を示す。

図１は、例えば、標準的な構成のワークステーションもしくはパーソナルコンピュータを利用して、ユーザレベルにおいて低オーバヘッドでＩ／Ｏ装置をアクセスする方式を実現するためのシステムである。

【００１９】

このシステムは、Ｉ／Ｏ装置（Ａ）１－１及びＩ／Ｏ装置（Ｂ）１－２、プロセッサ２、キャッシュ３、システムＬＳＩ４、メインメモリ５、Ｉ／Ｏバス６を備える。Ｉ／Ｏ装置（Ａ）１－１及びＩ／Ｏ装置（Ｂ）１－２は、内部に未処理

の要求が登録されていることを示すペンディングレジスタを備える。このペンディングレジスタは、メモリ空間にマップ（メモリマップ）されていて、プロセッサからアクセスされる（詳細は、後述する。）。プロセッサ２は、キャッシュ３と協調して、ＯＳに従い各種処理を実行する。キャッシュ３は、処理の高速化のために、プログラムやデータをコピーしておく高速メモリである。システムＬＳＩ４は、Ｉ／Ｏ装置（Ａ）１－１及びＩ／Ｏ装置（Ｂ）１－２、プロセッサ２、メインメモリ５の相互の間で、データの送受を制御する。メインメモリ５は、オペレーティングシステム（ＯＳ）、アプリケーションとしての各種プロセス、物理アドレス空間等のプログラム及びデータ等を記憶する。また、Ｉ／Ｏバス６は、拡張バスとして構成することができる。

【００２０】

つぎに、図２に、本発明に係るアクセス方法の説明図を示す。この図は、ペンディングレジスタのアドレスマップについて示したものである。

【００２１】

この例では、プロセス(process)Ａ～Ｃ２１０～２１２、物理アドレス空間(physical address space)２２、Ｉ／Ｏ装置１(I/0 device)が示される。Ｉ／Ｏ装置１は、ペンディングレジスタ(pending register)２３、ＤＭＡエンジン(DMA engine)２４、要求実行用のレジスタ(register)２５、内蔵メモリ２６を備える。また、プロセスＡ～Ｃ用ページ２７０～２７２(例えば、page for Ａ,Ｂ)及びプロセスＡ～Ｃ用、メモリページ２８０～２８２（page ０～２）が物理アドレス空間に用意され、各プロセスによる要求(requests)が要求格納領域２９０～２９２に記憶される。

【００２２】

図３に、本発明に係るアクセス方法のＯＳ及びプロセスについてのフローチャートを示す。また、図４に、本発明に係るアクセス方法のＩ／Ｏ装置についてのフローチャートを示す。以下、図２～図４を参照して、本発明のアクセス方法について説明する。

【００２３】

まず、図２及び図３のフローチャートに基づき、ＯＳ及びプロセス側に関する

処理について説明する。

ＯＳはプロセスを管理しているので、プロセスＡ～Ｃの存在をプロセスが作成された時から把握している。オープン処理によってわかることは、オープン処理をＯＳに要求したプロセスが、今後Ｉ／Ｏ装置１を使用しようとしていることである。まず、プロセスから見たＩ／Ｏ装置を使用するための処理が以下のように実行される。

【００２４】

ステップＳ１０１では、あるプロセス（例えば、プロセスＡ２１０）がＩ／Ｏ装置１のオープン処理をＯＳに要求する。このとき、プロセスは自分のＩ／Ｏ装置１への要求を置く領域をＯＳに指示する。つぎに、ステップＳ１０２では、第１に、ＯＳはオープン処理の中で、プロセスＡ２１０に対して使っていないＩ／Ｏ装置１用のコンテクストＩＤ（context-ID）（例えば、ＩＤ=0）を割り当て、そのＩＤに該当するメモリページ２８０（page 0）をプロセスＡ２１０用のペンディングレジスタ２３アクセス用のアドレスとしてマップする。マップした論理アドレスはオープン処理の結果としてプロセスＡ２１０にＯＳから通知される。また、ステップＳ１０２では、第２に、ＯＳが、オープン処理の中で、Ｉ／Ｏ装置１内の内蔵メモリ２６にプロセスＡ２１０の要求格納領域２９０へのポインタ（物理アドレス）を記憶する。記憶するアドレスはプロセスＡ２１０に割り当てたコンテクストＩＤから容易に計算可能な場所にする。つまり、コンテクストＩＤから要求格納領域へのポインタが取り出せる表が、内蔵メモリ上に形成される。

【００２５】

ステップＳ１０３では、プロセスＡ２１０は自分の要求格納領域２９０へＩ／Ｏ装置１への要求内容を記述する。ステップＳ１０４では、要求内容の記述後、ＯＳは、オープン処理で割り当てられたペンディングレジスタ２３用のアドレスを使って、未処理の要求があることをＩ／Ｏ装置に伝える。具体的にはペンディングレジスタ２３のアドレスに対してアクセスする。ステップＳ１０５では、Ｉ／Ｏ装置が要求格納領域２９０から要求を読み出す。ステップＳ１０６により、プロセスＡ２１０がＩ／Ｏ装置１に対して再び要求がある場合は、ステップＳ１

０３及びＳ１０４を繰り返す。ステップＳ１０７では、プロセスＡがＩ／Ｏ装置１のクローズ処理をＯＳに要求した場合、又はプロセスＡ２１０が終了した場合等には、ＯＳはプロセスＡ２１０に割り当てたアドレスを回収すると共に、Ｉ／Ｏ装置用のコンテクストＩＤの回収（空きＩＤとして記憶）及び内蔵メモリ２６上のプロセスＡ２１０の要求格納領域２９０へのポインタのエントリをクリアする。

【００２６】

つぎに、図２及び図４のフローチャートに基づき、Ｉ／Ｏ装置側に関する処理について説明する。Ｉ／Ｏ装置側から見ると、以下のように処理が実行される。

【００２７】

ステップＳ２０１では、ペンディングレジスタ２３にアクセスがあったことが、先入れ先出しメモリ（FIFO）出力からわかる。ステップＳ２０２では、FIFO出力にはコンテクストＩＤが含まれており、内蔵メモリ２６上の表を参照することにより、ペンディングレジスタをアクセスしたプロセスＡ２１０の要求格納領域２９０の物理アドレスがわかる。ステップＳ２０３では、ＤＮＡ エンジン２４により要求格納領域２９０の内容を読み出して、内容に沿って内部レジスタ２５を制御して、要求を実現する。ステップＳ２０４により、例えば、電源断まで、この動作をＩ／Ｏ装置は繰り返す。

【００２８】

プロセスＢもプロセスＣもまったく同様にしてＩ／Ｏ装置１を使用する。ここで、ステップＳ１０３、Ｓ１０４においてＯＳを一切介していないことがオーバヘッド低減の観点で重要である。たとえ、ホストプロセッサで動いているプロセスが時分割で切り替わっても、コンテクストＩＤを含むペンディングレジスタへのアドレスを使うことによって、一意に要求しているプロセスがわかるので、プロセス切替え時にＯＳは一切付加的な動作が必要ないことになる。なお、従来方式３ではコンテクストレジスタを切替える必要があった。

【００２９】

つぎに、図５に、ペンディングレジスタに記憶する機構の説明図を示す。

図５の一番上のメモリ構造は、Ｉ／Ｏ装置を指定する物理アドレス５１を示す

ここでは、一例として、36bit構成で０～１１ビットにページオフセット(Page Offset)としてのファンクションセレクト（Function Select)、１２～１９ビットにプロセスのためのコンテクストＩＤ(Context-ID)、２０～３５ビットにファンクションセレクト(Function Select)の各領域を含む。また、図のように、ページオフセット（大抵のプロセッサでは12bitか13bit）より上位の部分に、コンテクストＩＤを示すフィールドを用意する（図では8bit）。ただし、コンテクストＩＤのフィールドは、アドレスデコードに使用されず、ペンディングレジスタのアドレスデコードに使用され得るのはコンテクストＩＤよりも上位のフィールドとページオフセットの部分である。ファンクションセレクト領域は、ペンディングレジスタの位置を示す。

【００３０】

ファンクションセレクトは上位アドレスに関しては必ずデコードする（一意に決める）必要があり、一方、下位アドレス（オフセット部分）は、必ずしもデコードされなくても良い。オフセット部分をデコード対象とならないようにした場合、オフセット部分がどんなパターンになっていても上位アドレスのみによりペンディングレジスタにアクセスすることになる。なお、ページ単位でしかプロセスにアドレスを割り当てることはできないので、オフセット部のデコードを省略しても他のプロセスには影響がでない。逆に省略しないと、ペンディングレジスタ以外のレジスタやメモリをペンディングレジスタと同一のページ内に割り当てることができる。この場合、ページオフセット部分のビットパターンが違うアドレスが使われる。

【００３１】

また、コンテクストＩＤ領域は、Ｉ／Ｏ装置を使用するプロセス（例えば、プロセスＡ～Ｃのいずれか）を示す。コンテクストＩＤ領域が8ビットであれば、同時に256のプロセスがＩ／Ｏ装置の使用を要求（オープン）していても対応することができる。

【００３２】

ペンディングレジスタ用のアドレスデコーダ５２は、物理アドレス５１のファ

ンクションセレクト領域にペンディングレジスタの場所を表す固有アドレスが記憶されている場合、ペンディングレジスタを選択するとともに、物理アドレス５１のコンテクストＩＤをペンディングレジスタ５３に記憶する。また、必要に応じて、ペンディングレジスタ５３に、プロセス識別子に対応してデータ５４を記憶することができる。データ５４は、例えば、プロセッサ、メインメモリ、インタフェース等から与えられることができる（例えば、ストア・スワップアクセス時）。ペンディングレジスタ５３の出力５５には、コンテクストＩＤが含まれ、場合によってはデータもそれに対応して含まれる。

【００３３】

各アプリケーションは、特定のコンテクストＩＤを割り当てられ、そのＩＤに従ったページのみをマップする。つまり、アプリケーションが異なれば、コンテクストＩＤの部分が異なったアドレスで同一のペンディングレジスタへアクセスすることになる。Ｉ／Ｏ装置側にアクセスされるたびにコンテクストＩＤ部分を保持して解釈する能力があれば、どのアプリケーションがペンディングレジスタへアクセスしたか明らかにすることができる。ペンディングレジスタへのアクセスは、ユーザアプリケーションによるアクセスであり、実行されるタイミングに制約がない。Ｉ／Ｏ装置の動作とペンディングレジスタによる新規要求の受け付けは同期されないと入出力処理が行えなくなる。このため、ペンディングレジスタ５３へのアクセスをそれを構成するFIFOにバッファリングしてやる必要がある。このときにアクセス内容（書き込みなら「書き込み」であるという情報と「書き込みデータ」）のみではなく、コンテクストＩＤも同時にバッファリングする。コンテクストＩＤから適当な処理によってアプリケーションの要求内容を格納しているアドレス空間とアドレスを求めて、Ｉ／Ｏ装置が要求内容を読み出し、入出力処理を行う。

【００３４】

FIFOバッファの溢れに関しては、以下のように、返り値もしくは割り込みで対処することができる。

・返り値で対処する場合は、「書き込みデータ」が不要であれば、l ad（メモリ読み出し）命令によってペンディングレジスタをアクセスし、FIFOバッファが溢

れている場合には返り値としてエラーコード（例えば-1）を返す。ユーザアプリケーションは、返り値を見て、成功するまでアクセスを繰り返す。「書き込みデータ」が必要であれば、swap（不可分読み書き）命令によってを返り値を受け取りつつ、データをＩ／Ｏ装置１に受け渡す。

・Ｉ／Ｏ装置用の拡張バスを介したload命令やswap命令は、store命令と比べるとプロセッサにとって実行時間が長くコストが高い。そこで、割り込みで対処する場合は、store命令によってペンディングレジスタへアクセス可能にするためには、FIFOバッファ溢れを割り込みによって検出する。FIFOバッファの容量が十分にあり、Ｉ／Ｏ装置１の能力が飽和していなければ、FIFOバッファ溢れは発生しない。FIFOバッファ溢れが発生した場合には、処理コストが少しぐらい掛かってもやむを得ないと考えて、割り込みを発生させてペンディングレジスタへアクセスに失敗したアプリケーションやアクセス内容をＯＳによって保管しておき、Ｉ／Ｏ装置に登録可能になった時点で処理する。

【００３５】

なお、本発明に係るアクセス方法は、Ｉ／Ｏ装置に限らず、適宜のインタフェースのアクセスに適用することができる。また、本発明に係るアクセス方法は、アクセス処理プログラムとして、CD-ROM等の記録媒体又はインタネット等の伝送媒体により提供されることができる。

【００３６】

【発明の効果】

本発明によると、以上のように、メモリ管理機構を流用することにより、複数のアプリケーションが同時に低コストで通信および入出力操作、インタフェース操作を可能とするアクセス方法及びアクセス処理プログラムを記録した記録媒体を提供することができる。

【図面の簡単な説明】

【図１】

本発明に関連するシステム構成図。

【図２】

本発明に係るアクセス方法の説明図。

【図３】

本発明に係るアクセス方法のＯＳ及びプロセスについてのフローチャート。

【図４】

本発明に係るアクセス方法のＩ／Ｏ装置についてのフローチャート。

【図５】

ペンディングレジスタに記憶する機構の説明図。

【図６】

従来方式１のシステムの説明図。

【図７】

従来方式２のシステムの説明図。

【図８】

従来方式３のシステムの説明図。

【符号の説明】

１　Ｉ／Ｏ装置

２　プロセッサ

３　キャッシュ

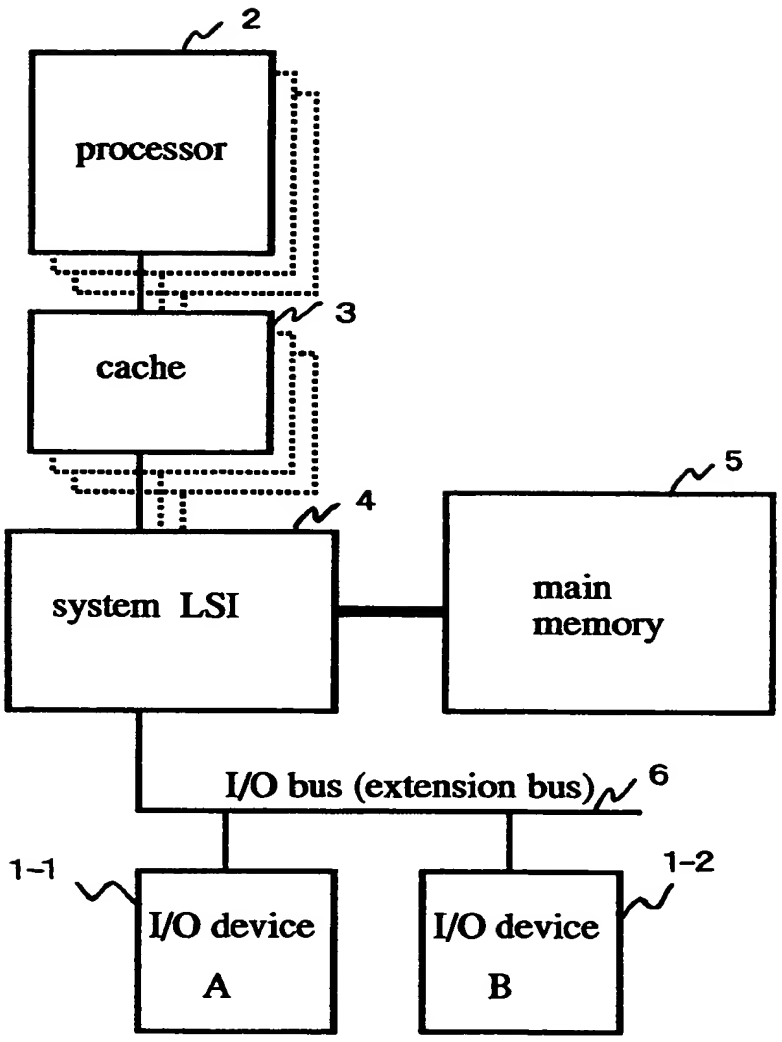
４　システムＬＳＩ

５　メインメモリ５

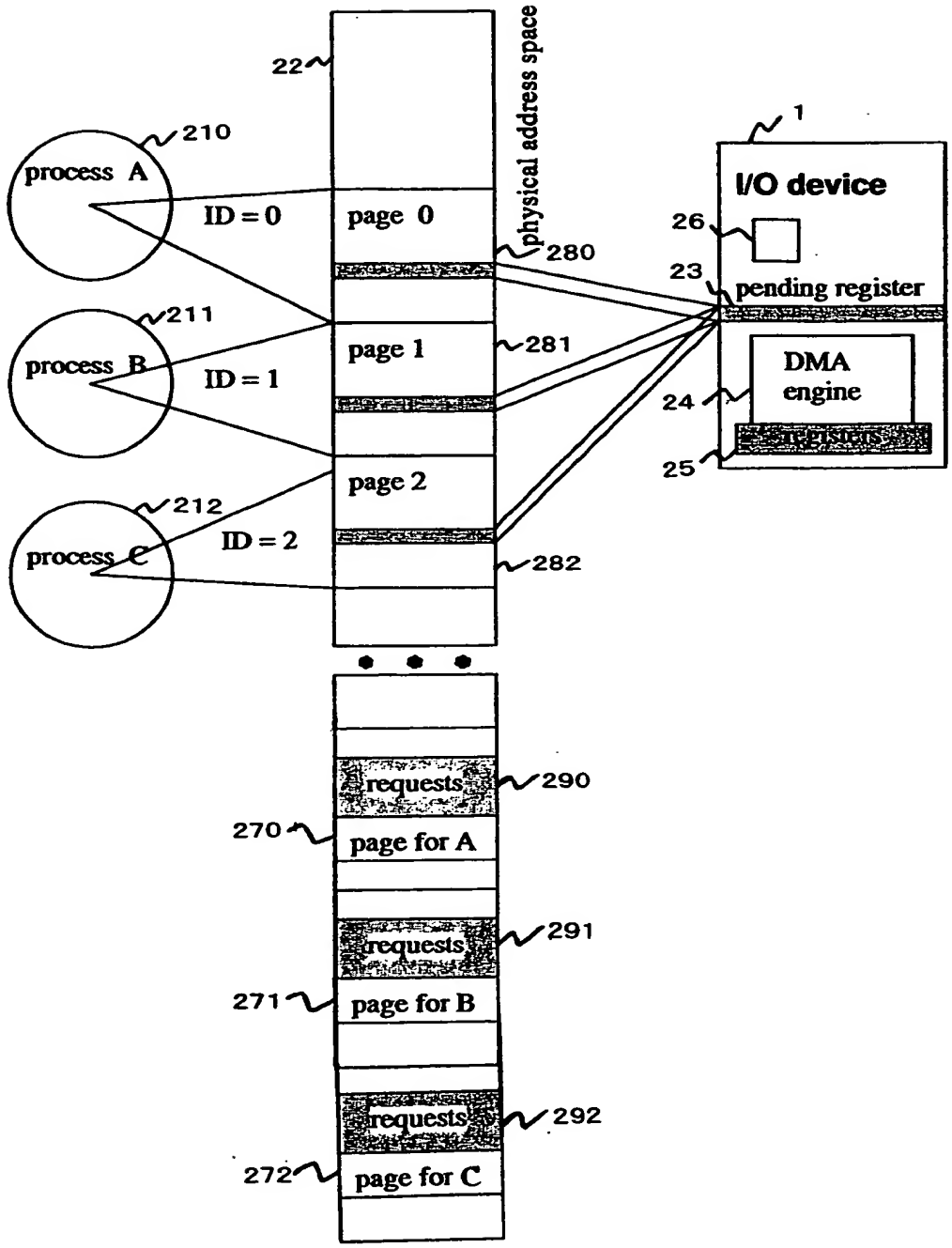
６　Ｉ／Ｏバス

【書類名】図面

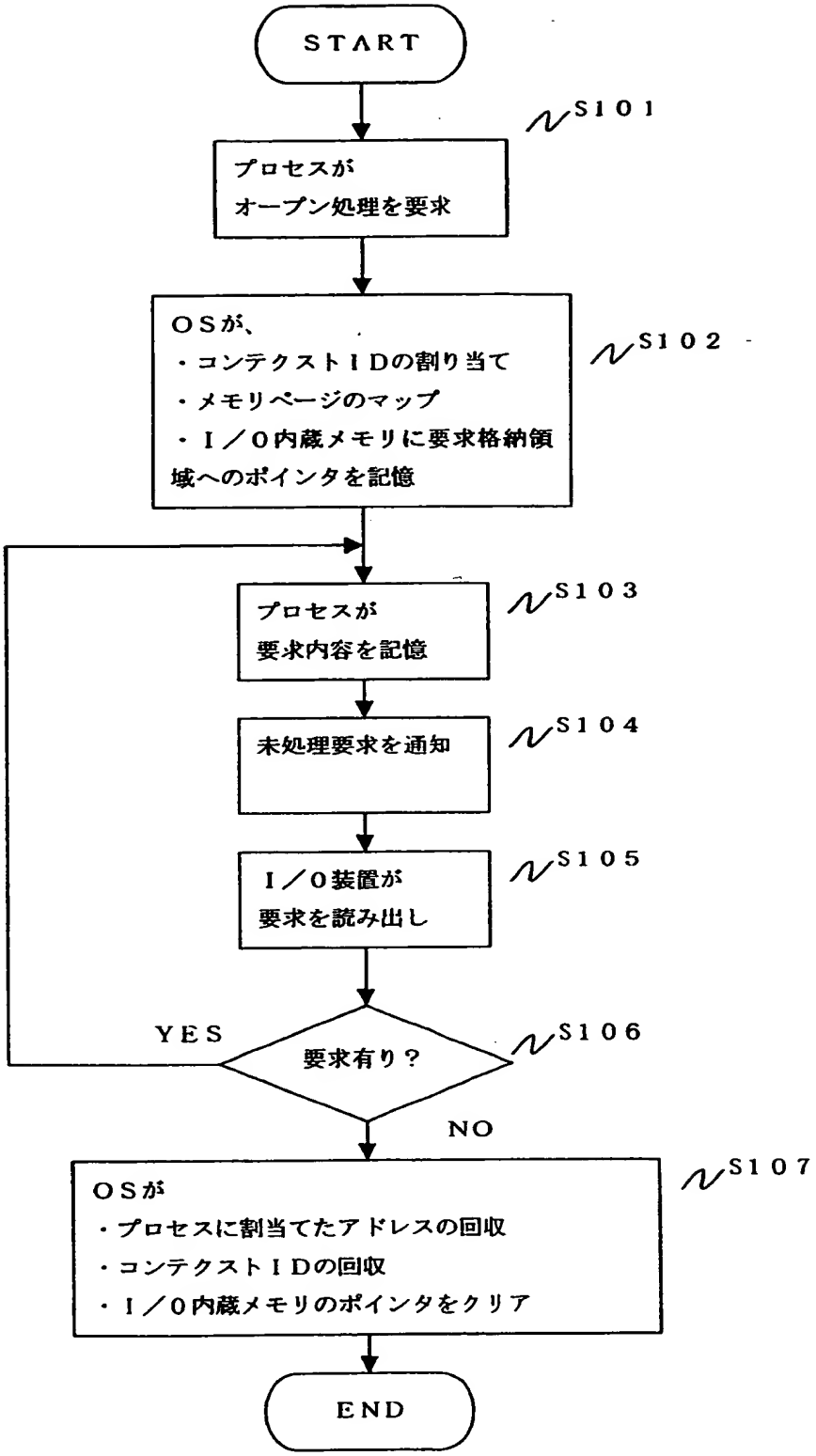
【図１】


2
processor
3
cache
4
system LSI
5
main
memory
I/O bus (extension bus)
6
1-1
I/O device
A
1-2
I/O device
B

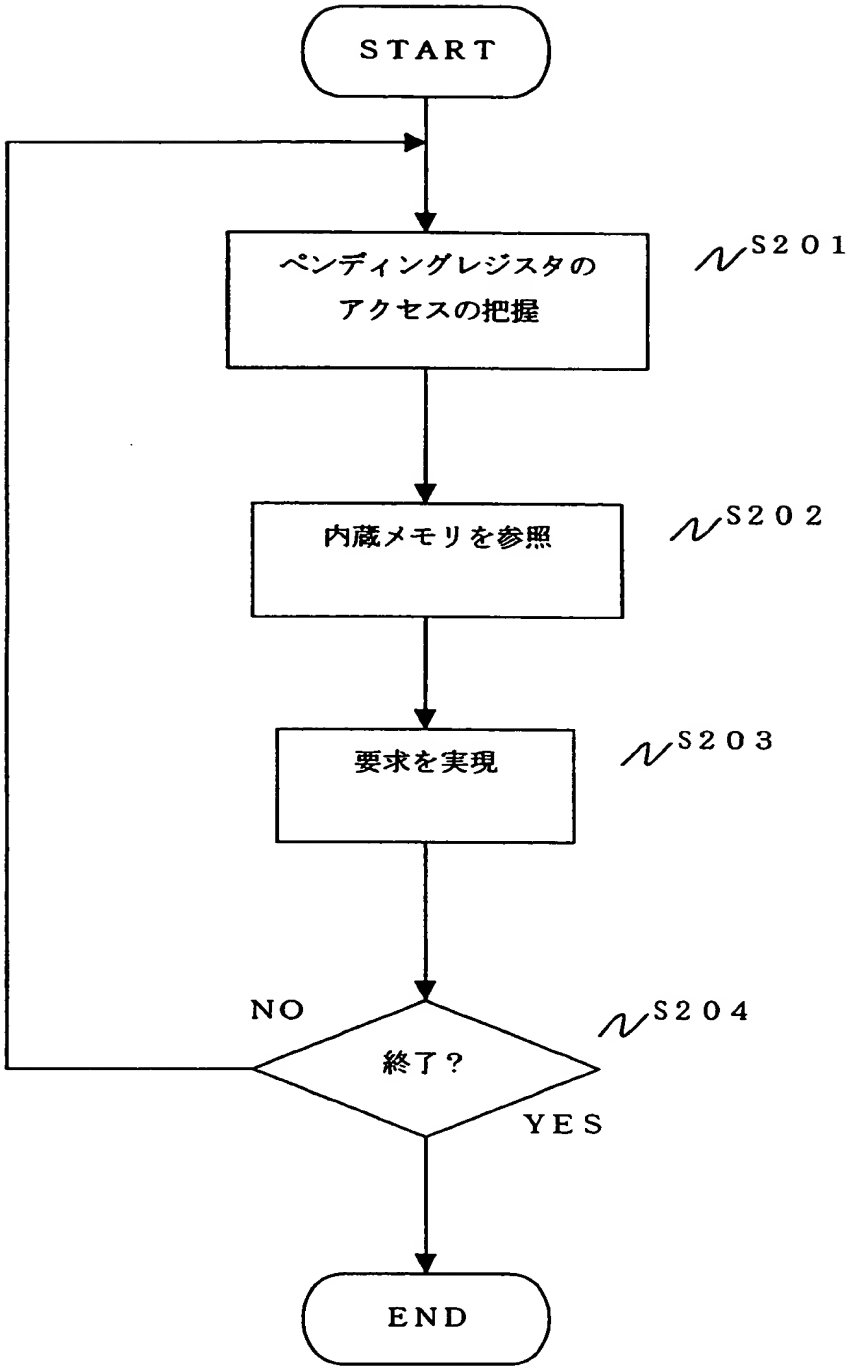
【図２】


22
physical address space
210
process A
ID = 0
page 0
280
1
I/O device
26
23
pending register
211
process B
ID = 1
page 1
281
DMA
engine
24
registers
25
page 2
212
process C
ID = 2
282
requests
290
270
page for A
requests
291
271
page for B
requests
292
272
page for C

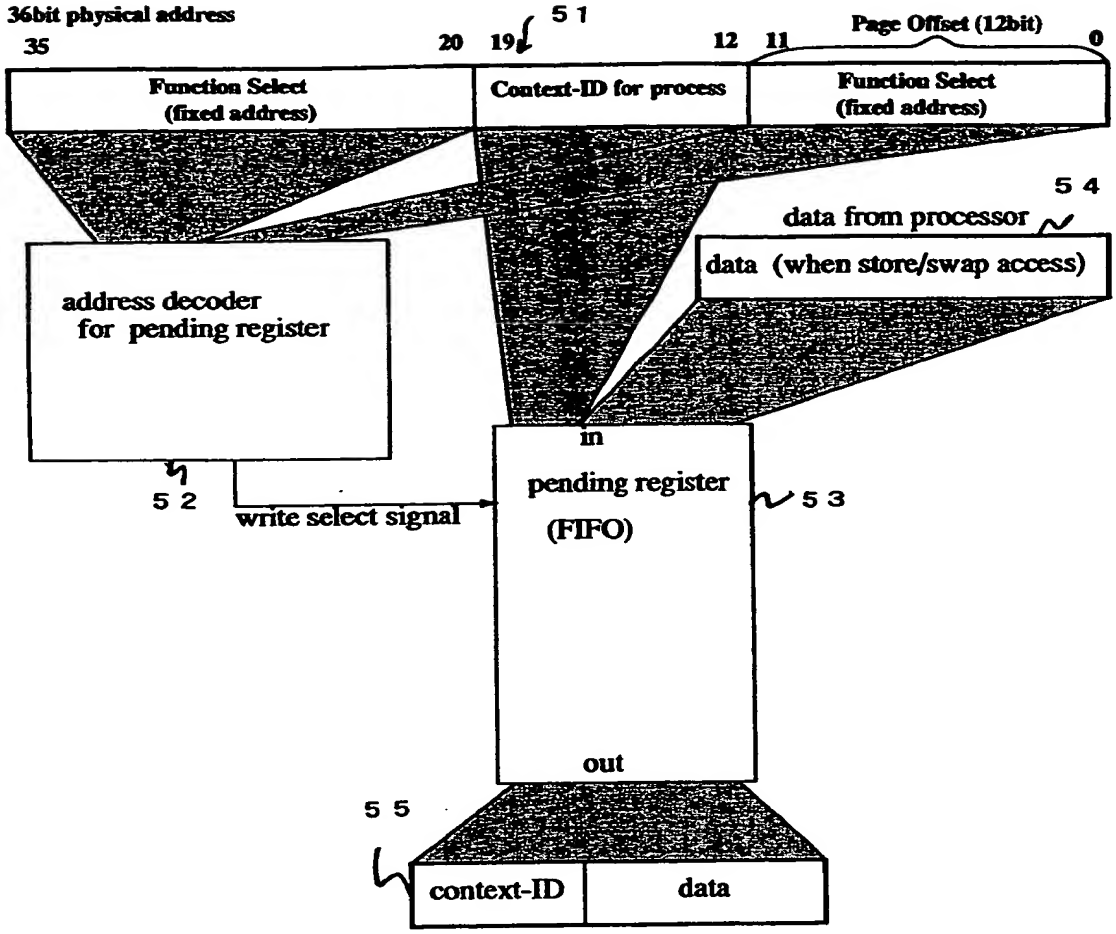
【図３】


START
S101
プロセスが
オープン処理を要求
S102
OSが、
・コンテクストＩＤの割り当て
・メモリページのマップ
・Ｉ／Ｏ内蔵メモリに要求格納領域へのポインタを記憶
S103
プロセスが
要求内容を記憶
S104
未処理要求を通知
S105
Ｉ／Ｏ装置が
要求を読み出し
S106
YES
要求有り？
NO
S107
OSが
・プロセスに割当てたアドレスの回収
・コンテクストＩＤの回収
・Ｉ／Ｏ内蔵メモリのポインタをクリア
END

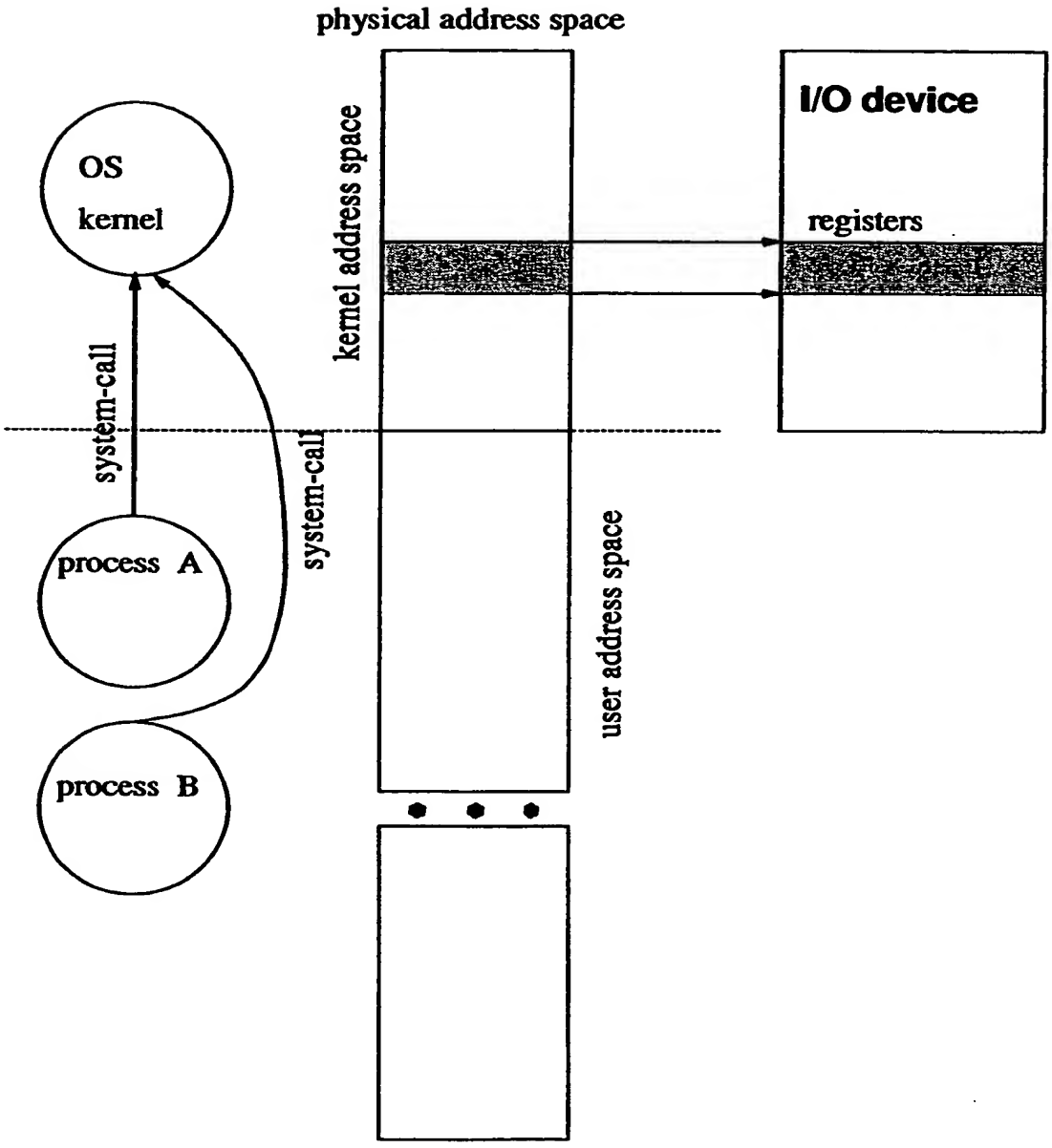
【図４】


START
ペンディングレジスタの
アクセスの把握
S２０１
内蔵メモリを参照
S２０２
要求を実現
S２０３
NO
終了？
S２０４
YES
END

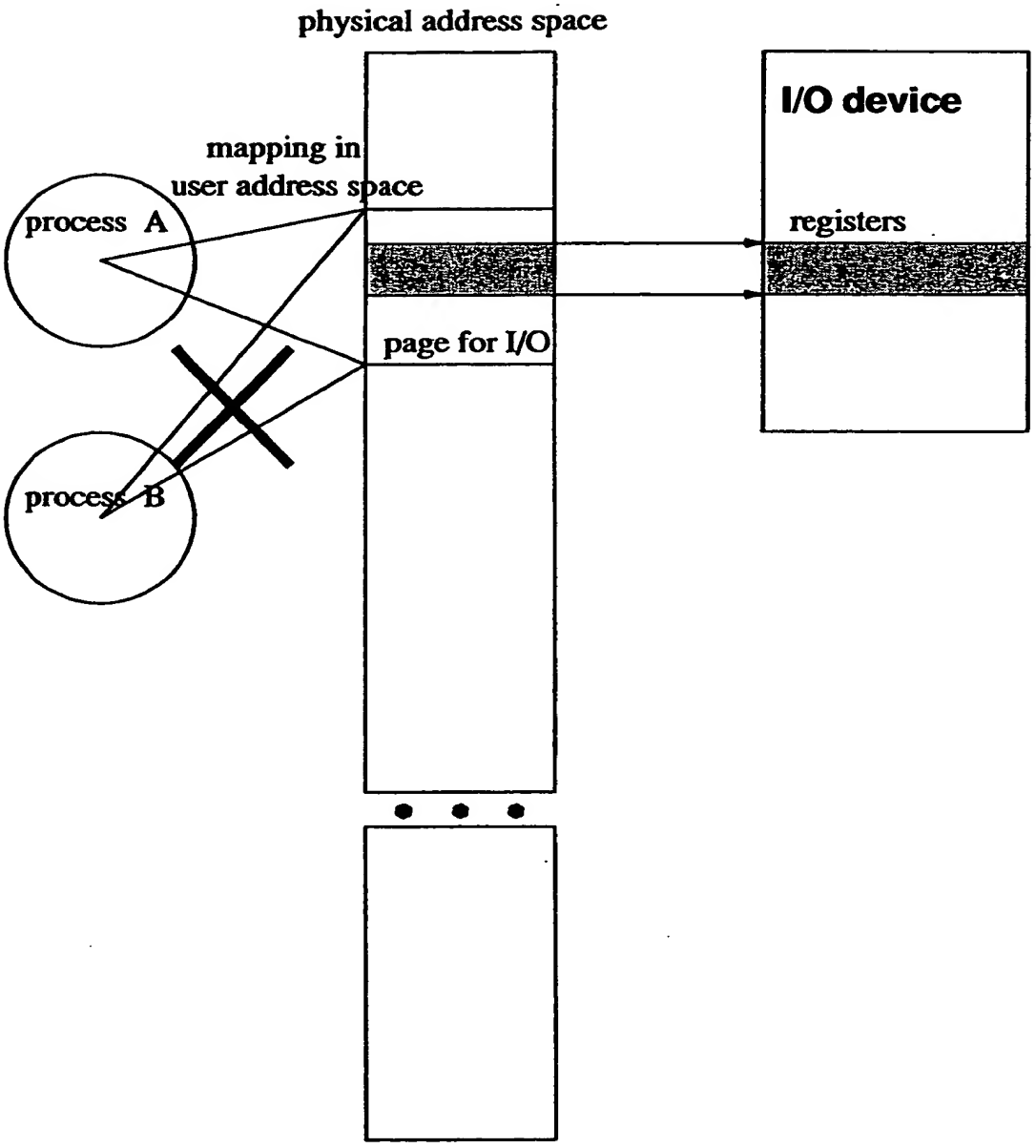
【図５】


36bit physical address
35
20
19
51
12
11
Page Offset (12bit)
0
Function Select
(fixed address)
Context-ID for process
Function Select
(fixed address)
54
data from processor
data (when store/swap access)
address decoder
for pending register
52
write select signal
in
pending register
(FIFO)
53
out
55
context-ID
data

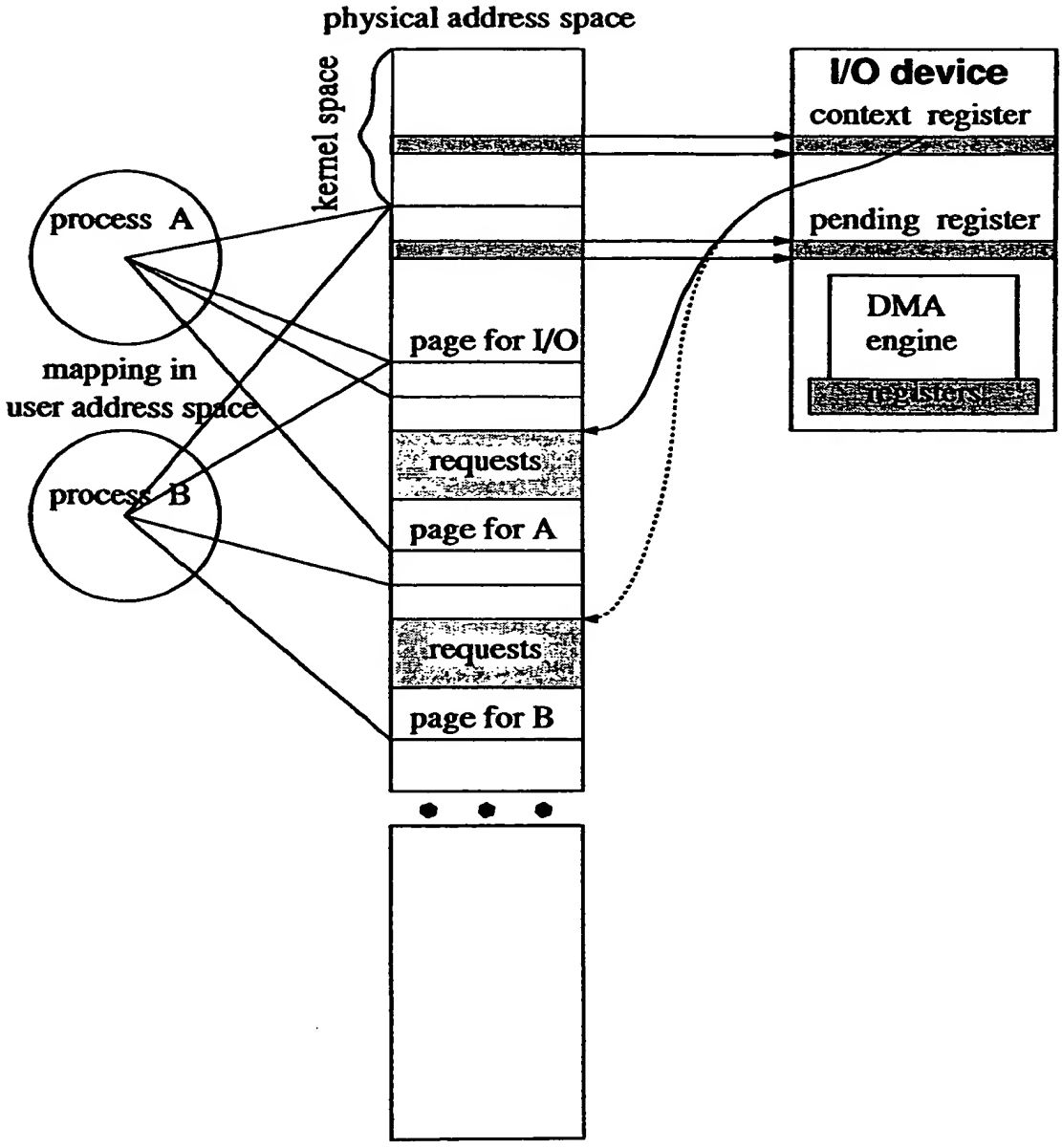
【図６】


physical address space
I/O device
registers
kernel address space
OS
kernel
system-call
system-call
process A
process B
user address space

【図７】


physical address space
I/O device
mapping in
user address space
process A
registers
page for I/O
process B

【図８】


physical address space
kernel space
I/O device
context register
pending register
DMA
engine
registers
process A
mapping in
user address space
process B
page for I/O
requests
page for A
requests
page for B

【書類名】　　要約書

【要約】

【課題】　複数のアプリケーションが同時に低コストで通信および入出力操作、インタフェース操作を可能とする。

【解決手段】　プロセスＡがＩ／Ｏ装置への要求を置く領域をＯＳに指示する。ＯＳは、プロセスＡに対して使っていないＩ／Ｏ装置用のコンテクストＩＤを割り当て、そのＩＤに該当するメモリページをプロセスＡ用のペンディングレジスタアクセス用のアドレスとしてマップし、Ｉ／Ｏ装置内の内蔵メモリにプロセスＡの要求格納領域へのポインタ（物理アドレス）を記憶する。プロセスＡは自分の要求格納領域へ要求内容を記述し、ＯＳは、ペンディングレジスタ用のアドレスを使って、未処理の要求があることをＩ／Ｏ装置に伝える。Ｉ／Ｏ装置は、ＤＭＡエンジンにより要求格納領域の内容を読み出して、要求を実現する。

【選択図】　図２

出　願　人　履　歴　情　報

識別番号　　〔３９６０２０８００〕

１．変更年月日　　１９９８年　２月２４日
［変更理由］　名称変更
住　所　　埼玉県川口市本町４丁目１番８号
氏　名　　科学技術振興事業団